NAME : Pratiksha Ranmare

ROLL NO.: 654

BATCH : F3

**ASSINGMENT 3**

import numpy as np

array1=np.array([[1,2,3],[4,5,6],[7,8,9]])

array1

OUTPUT:

array([[1, 2, 3],

[4, 5, 6],

[7, 8, 9]])

array2=np.array([[11,12,13],[14,15,16],[17,18,19]])

array2

OUTPUT:

array([[11, 12, 13],

[14, 15, 16],

[17, 18, 19]])

1. MATRIX OPERATION

1.1 ADDITION

resultarray=array1+array2

print("\nUsing Operator:\n",resultarray)

resultarray=np.add(array1,array2)

print("\nUsing Numpy Function:\n", resultarray)

OUTPUT:

Using Operator:

[[12 14 16]

[18 20 22]

[24 26 28]]

Using Numpy Function:

[[12 14 16]

[18 20 22]

[24 26 28]]

1.2 SUBTRACTION

resultarray=array1-array2

print("\nusing Operator:\n",resultarray)

resultarray=np.subtract(array1,array2)

print("\nUsing Numpy Fucntion:\n",resultarray)

OUTPUT:

using Operator:

[[-10 -10 -10]

[-10 -10 -10]

[-10 -10 -10]]

Using Numpy Fucntion:

[[-10 -10 -10]

[-10 -10 -10]

[-10 -10 -10]]

1.3 MULTIPLICATION

resultarray=array1\*array2

print("\nUsing Operator:\n",resultarray)

resultarray=np.multiply(array1,array2)

print("\nUsing Numpy Function:\n",resultarray)

OUTPUT:

Using Operator:

[[ 11 24 39]

[ 56 75 96]

[119 144 171]]

Using Numpy Function:

[[ 11 24 39]

[ 56 75 96]

[119 144 171]]

1.4 DIVISION

resultarray=array1/array2

print("\nUsing Operator:\n",resultarray)

resultarray=np.divide(array1,array2)

print("\nUsing Numpy Function:\n",resultarray)

OUTPUT:

Using Operator:

[[0.09090909 0.16666667 0.23076923]

[0.28571429 0.33333333 0.375 ]

[0.41176471 0.44444444 0.47368421]]

Using Numpy Function:

[[0.09090909 0.16666667 0.23076923]

[0.28571429 0.33333333 0.375 ]

[0.41176471 0.44444444 0.47368421]]

1.5 MOD

resultarray=array1%array2

print("\nUsing Operator:\n",resultarray)

resultarray=np.mod(array1,array2)

print("\nUsing Numpy Function:\n",resultarray)

OUTPUT:

Using Operator:

[[1 2 3]

[4 5 6]

[7 8 9]]

Using Numpy Function:

[[1 2 3]

[4 5 6]

[7 8 9]]

1.6 DOT PRODUCT

resultarray=np.dot(array1,array2)

print("\nUsing Numpy Function:\n",resultarray)

OUTPUT:

Using Numpy Function:

[[ 90 96 102]

[216 231 246]

[342 366 390]]

1.7 TRANSPOSE

resultarray=np.transpose(array1)

print(resultarray)

#Or

resultarray=array1.transpose()

print(resultarray)

OUTPUT:

[[1 4 7]

[2 5 8]

[3 6 9]]

[[1 4 7]

[2 5 8]

[3 6 9]]

## **2. HORIZONTAL AND VERTICAL STACKING OF NUMPY ARRAYS**

2.1 HORIZANTAL STACKING

resultarray=np.hstack((array1,array2))

resultarray

OUTPUT:

array([[ 1, 2, 3, 11, 12, 13],

[ 4, 5, 6, 14, 15, 16],

[ 7, 8, 9, 17, 18, 19]])

2.2 VERTICAL STACKING

resultarray=np.vstack((array1,array2))

resultarray

OUTPUT:

array([[ 1, 2, 3],

[ 4, 5, 6],

[ 7, 8, 9],

[11, 12, 13],

[17, 18, 19]])

## **3. CUSTOM SEQUENCE GENERATION**

3.1 RANGE

nparray=np.arange(0,12,1).reshape(3,4)

nparray

array([[ 0, 1, 2, 3],

[ 4, 5, 6, 7],

[ 8, 9, 10, 11]])

3.2 LINEARLY SEPARABLE

nparray=np.linspace(start=0,stop=24,num=12).reshape(3,4)

nparray

OUTPUT:

array([[ 0. , 2.18181818, 4.36363636, 6.54545455],

[ 8.72727273, 10.90909091, 13.09090909, 15.27272727],

[17.45454545, 19.63636364, 21.81818182, 24. ]])

3.3 EMPTY ARRAY

nparray=np.empty((3,3),int)

nparray

OUTPUT:

array([[ 90, 96, 102],

[216, 231, 246],

[342, 366, 390]])

3.4 EMPTY LIKE SOME OTHER ARRAY

nparray=np.empty\_like(array1)

nparray

OUTPUT:  
array([[ 90, 96, 102],

[216, 231, 246],

[342, 366, 390]])

3.5 IDENTITY MATRIX

nparray=np.identity(3)

nparray

OUTPUT:  
array([[1., 0., 0.],

[0., 1., 0.],

[0., 0., 1.]])

## **4. ARITHMETIC AND STATISTICAL OPERATIONS, MATHEMATICAL OPERATIONS, BITWISE OPERATIONS**

4.1 ARITHMETIC OPERATIONS

array1=np.array([1,2,3,4,5])

array2=np.array([11,12,13,14,15])

print(array1)

print(array2)

OUTPUT:

[1 2 3 4 5]

[11 12 13 14 15]

# Addition

print(np.add(array1,array2))

# Subtraction

print(np.subtract(array1,array2))

# Multiplication

print(np.multiply(array1,array2))

# Division

print(np.divide(array1,array2))

[12 14 16 18 20]

[-10 -10 -10 -10 -10]

[11 24 39 56 75]

[0.09090909 0.16666667 0.23076923 0.28571429 0.33333333]

4.2 STATISTICAL OPERATIONS

array1=np.array([1,2,3,4,5,9,6,7,8,9,9])

# Standard Deviation

print(np.std(array1))

#Minimum

print(np.min(array1))

#Summation

print(np.sum(array1))

#Median

print(np.median(array1))

#Mean

print(np.mean(array1))

#Mode

from scipy import stats

print("Most Frequent element=",stats.mode(array1)[0])

print("Number of Occurances=",stats.mode(array1)[1])

#Variance

print(np.var(array1))

2.7990553306073913

1

63

6.0

5.7272727272727275

Most Frequent element= [9]

Number of Occurances= [3]

7.834710743801653

4.3 BITWISE OPERATIONS

array1=np.array([1,2,3],dtype=np.uint8)

array2=np.array([4,5,6])

# AND

resultarray=np.bitwise\_and(array1,array2)

print(resultarray)

# OR

resultarray=np.bitwise\_or(array1,array2)

print(resultarray)

#LeftShift

resultarray=np.left\_shift(array1,2)

print(resultarray)

#RightShift

resultarray=np.right\_shift(array1,2)

print(resultarray)

OUTPUT:

[0 0 2]

[5 7 7]

[ 4 8 12]

[0 0 0]

## You can get Binary Representation of Number ######

print(np.binary\_repr(10,8))

resultarray=np.left\_shift(10,2)

print(resultarray)

print(np.binary\_repr(np.left\_shift(10,2),8))

OUTPUT:

00001010

40

00101000

## **5.COPYING AND VIEWING ARRAYS**

5.1 COPY

array1=np.arange(1,10)

print(array1)

newarray=array1.copy()

print(newarray)

##modification in Original Array

array1[0]=100

print(array1)

print(newarray)

OUTPUT:

[1 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

[100 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

5.2 VIEW

array1=np.arange(1,10)

print(array1)

newarray=array1.view()

print(newarray)

##modification in Original Array

array1[0]=100

print(array1)

print(newarray)

OUTPUT:

[1 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

[100 2 3 4 5 6 7 8 9]

[100 2 3 4 5 6 7 8 9]

## **6. SEARCHING**

array1=np.array([[1,2,3,12,5,7],[94,5,6,7,89,44],[7,8,9,11,13,14]])

print(array1)

OUTPUT:  
[[ 1 2 3 12 5 7]

[94 5 6 7 89 44]

[ 7 8 9 11 13 14]]

6.1 HORRIZANTALLY SORT

np.sort(array1,axis=0)

OUTPUT:

array([[ 1, 2, 3, 7, 5, 7],

[ 7, 5, 6, 11, 13, 14],

[94, 8, 9, 12, 89, 44]])

6.2 VERTICALLY SORT

np.sort(array1,axis=1)

OUTPUT:  
array([[ 1, 2, 3, 5, 7, 12],

[ 5, 6, 7, 44, 89, 94],

[ 7, 8, 9, 11, 13, 14]])

## **7.SEARCHING**

import numpy as np

array1 =np.array([1,2,3,12,5,7])

np.searchsorted(array1,7,side="left")#Perform Search After sorting

OUTPUT:3

## **8.COUNTING**

array1=np.array([1,2,3,12,5,7,0])

print(np.count\_nonzero(array1))#Return total Non Zero element

print(np.nonzero(array1))#Return Index

print(array1.size)#Total Element

OUTPUT:

6

(array([0, 1, 2, 3, 4, 5]),)

7

## **9. DATA STACKING**

array1=np.array(np.arange(1,5).reshape(2,2))

print(array1)

array2=np.array(np.arange(11,15).reshape(2,2))

print(array2)

OUTPUT:  
[[1 2]

[3 4]]

[[11 12]

[13 14]]

newarray=np.stack([array1,array2],axis=0)

print(newarray)

OUTPUT:

[[[ 1 2]

[ 3 4]]

[[11 12]

[13 14]]]

newarray=np.stack([array1,array2],axis=1)

print(newarray)

OUTPUT:

[[[ 1 2]

[11 12]]

[[ 3 4]

[13 14]]]

## **10. APPEND**

array1=np.arange(1,10).reshape(3,3)

print(array1)

array2=np.arange(21,30).reshape(3,3)

print(array2)

OUTPUT:  
[[1 2 3]

[4 5 6]

[7 8 9]]

[[21 22 23]

[24 25 26]

[27 28 29]]

np.append(array1,array2,axis=0)

OUTPUT:  
array([[ 1, 2, 3],

[ 4, 5, 6],

[ 7, 8, 9],

[21, 22, 23],

[24, 25, 26],

[27, 28, 29]])

np.append(array1,array2,axis=1)

OUTPUT:

array([[ 1, 2, 3, 21, 22, 23],

[ 4, 5, 6, 24, 25, 26],

[ 7, 8, 9, 27, 28, 29]])

## **11.CONCATINATE**

array1=np.arange(1,10).reshape(3,3)

print(array1)

array2=np.arange(21,30).reshape(3,3)

print(array2)

OUTPUT:

[[1 2 3]

[4 5 6]

[7 8 9]]

[[21 22 23]

[24 25 26]

[27 28 29]]

np.concatenate((array1,array2),axis=0)

OUTPUT:

array([[ 1, 2, 3

[ 4, 5, 6],

[ 7, 8, 9],

[21, 22, 23],

[24, 25, 26],

[27, 28, 29]])

np.concatenate((array1,array2),axis=1)

OUTPUT:

array([[ 1, 2, 3, 21, 22, 23],

[ 4, 5, 6, 24, 25, 26],

[ 7, 8, 9, 27, 28, 29]])